

WHAT IS CLAIMED:

- 1           1.       A method comprising:  
2                    translating a first portion of subject code into a portion of target code;  
3                    caching said portion of target code; and  
4                    retrieving the cached portion of target code upon compatibility detection  
5                    between said portion of target code and a second portion of subject code.
- 1           2.       The method of claim 1 wherein compatibility of cache translations and  
2                    subject code to be translated is determined by cache key comparison.
- 1           3.       The method of claim 2 wherein the cache key is the byte sequence that  
2                    encodes the corresponding subject code instruction sequence.
- 1           4.       The method of claim 2 wherein the cache key is a hash of the  
2                    corresponding subject code instruction sequence.
- 1           5.       The method of claim 2 wherein the cache key comprises: (1) filename of  
2                    executable; (2) offset and length of the subject code sequence; (3) last modification time  
3                    of file; (4) version number of the translator; and (5) subject memory address of subject  
4                    code sequence.
- 1           6.       The method of claim 2 wherein the cache key comprises a plurality of  
2                    metrics.
- 1           7.       The method of claim 2 wherein compatibility is determined by computing  
2                    a cache key data structure corresponding to the subject code to be translated to a plurality

3 of second data structures, each second data structure corresponding to a different set of  
4 cached target code instructions.

1 8. The method of claim 1 further including the step of executing the target  
2 code.

1 9. The method of claim 1 wherein translations of self-modifying code are not  
2 cached.

1 10. The method of claim 1 wherein the portion of target code cached  
2 comprises a translation structure including a basic block.

1 11. The method of claim 1 wherein the portion of target code cached  
2 comprises one or more block translations and their respective successor lists.

1 12. The method of claim 1 wherein the portion of target code is converted into  
2 a single cache unit comprising a subject program and all its associated libraries.

1 13. The method of claim 1 wherein the portion of target code cached consists  
2 of a single instruction.

1 14. The method of claim 1 wherein the portion of target code cached  
2 comprises all code blocks corresponding to the same starting subject address.

1 15. The method of claim 1 wherein the portion of target code cached  
2 comprises a cache unit representing a discrete range of subject addresses.

1           16.     The method of claim 1 wherein the portion of target code cached as a unit  
2     comprises a subject library.

1           17.     In combination:  
2                 a target processor; and  
3                 translator code for translating subject program code into target code  
4     executable on said target processor, said translator code comprising code  
5     executable by said target processor to:  
6                 translate a first portion of subject code into a portion of target code;  
7                 cache said portion of target code; and  
8                 retrieve the cached portion of the target code upon detection of  
9     compatibility between said portion of target code and a second portion of subject  
10    code.

1           18.     The combination of claim 17 wherein compatibility of cached translations  
2     and subject code to be translated is determined by cache key comparison.

1           19.     The combination of claim 18 wherein the cache key is the byte sequence  
2     that encodes the corresponding subject code instruction sequence.

1           20.     The combination of claim 18 wherein the cache key is a hash of the  
2     corresponding subject code instruction sequence.

1           21.     The combination of claim 18 wherein the cache key comprises: (1) an  
2     identifier of the file containing the portion of subject code; (2) the offset and length of the  
3     subject code sequence; (3) last modification time of the file; (4) version number of the  
4     translator; and (5) subject memory address of the subject code sequence.

1           22.    The combination of claim 18 wherein the cache key comprises a plurality  
2 of metrics.

1           23.    The combination of claim 18 wherein compatibility is determined by  
2 comparing a cache key data structure corresponding to the subject code to be translated to  
3 a plurality of second data structures, each second data structure corresponding to a  
4 different set of cached target code instructions.

1           24.    The combination of claim 17 further including the step of executing the  
2 target code.

1           25.    The combination of claim 17 wherein translations of self-modifying code  
2 are not cached.

1           26.    The combination of claim 17 wherein the portion of target code cached  
2 comprises a translation structure including a basic block.

1           27.    The combination of claim 17 wherein the portion of target code cached  
2 comprises one or more block translations and their respective successor lists.

1           28.    The combination of claim 17 wherein the portion of target code is  
2 converted into a single cache unit comprising a subject program and all its associated  
3 libraries.

1           29.    The combination of claim 17 wherein the portion of target code cached  
2 consists of a single instruction.

1           30.    The combination of claim 17 wherein the portion of target code cached  
2 comprises all code blocks corresponding to the same starting subject address.

1           31.    The combination of claim 17 wherein the portion of target code cached  
2 comprises a cache unit representing a discrete range of subject addresses.

1           32     The combination of claim 17 wherein the portion of target code cached as  
2 a unit comprises a subject library.

1           33.    A program storage medium storing translator code for translating subject  
2 program code into target code, said translator code, when executed by a computer, being  
3 operable to perform the steps comprising:

4                   translating a first portion of subject code into a portion of target code;  
5                   caching said portion of target code; and  
6                   retrieving the cached portion of target code upon compatibility detection  
7           between said portion of target code and a second portion of subject code.

1           34.    The storage medium of claim 33 wherein compatibility of cache  
2 translations and subject code to be translated is determined by cache key comparison.

1           35.    The storage medium method of claim 34 wherein the cache key is the byte  
2 sequence that encodes the corresponding subject code instruction sequence.

1           36.    The storage medium of claim 34 wherein the cache key is a hash of the  
2 corresponding subject code instruction sequence.

1           37.     The storage medium of claim 34 wherein the cache key comprises: (1)  
2     filename of executable; (2) offset and length of the subject code sequence; (3) last  
3     modification time of file; (4) version number of the translator; and (5) subject memory  
4     address of subject code sequence.

1           38.     The storage medium of claim 33 wherein the cache key comprises a  
2     plurality of metrics.

1           39.     The storage medium of claim 34 wherein compatibility is determined by  
2     computing a cache key data structure corresponding to the subject code to be translated  
3     and comparing that data structure to a plurality of second data structures, each second  
4     data structure corresponding to a different set of cached target code instructions.

1           40.     The storage medium of claim 33 further including the step of executing  
2     the target code.

1           41.     The storage medium of claim 33 wherein translations of self-modifying  
2     code are not cached.

1           42.     The storage medium of claim 33 wherein the portion of target code cached  
2     comprises a translation structure including a basic block.

1           43.     The storage medium of claim 33 wherein the portion of target code cached  
2     comprises one or more block translations and their respective successor lists.

1           44.     The storage medium of claim 33 wherein the portion of target code is  
2 converted into a single cache unit comprising a subject program and all its associated  
3 libraries.

1           45.     The storage medium of claim 33 wherein the portion of target code cached  
2 consists of a single instruction.

1           46.     The storage medium of claim 33 wherein the portion of target code cached  
2 comprises all code blocks corresponding to the same starting subject address.

1           47.     The storage medium of claim 33 wherein the portion of target code cached  
2 comprises a cache unit representing a discrete range of subject addresses.

1           48.     The storage medium of claim 33 wherein the portion of target code cached  
2 as a unit comprises a subject library.

1           49.     In combination:  
2                   program code for translating a first portion of subject code into a portion  
3 of target code; and  
4                   program code for caching said portion of target code and for retrieving  
5 said target code upon detection of compatibility between a second portion of  
6 subject code and said portion of target code.

1           50.     The method of claim 1 wherein the first portion of subject code is part of a  
2 first program and the second portion of subject code is part of a second program.

1           51.     The method of claim 50 wherein said target code is cached at the end of  
2 translation of said first program.